Finding Misplaced Items in Retail by Clustering RFID Data

Leonardo Weiss Ferreira Chaves
SAP Research CEC Karlsruhe
Vincenz-Prießnitz-Str. 1
76131 Karlsruhe, Germany
leonardo.weiss.f.chaves@sap.com

Erik Buchmann, Klemens Böhm
Universität Karlsruhe (TH)
Am Fasanengarten 5
76131 Karlsruhe, Germany
{buchmann|boehm}@ipd.uka.de

ABSTRACT

In retail, products are organized according to layout plans, so-called planograms. Compliance to planograms is important, since good product placement can significantly increase sales. Currently, retailers are about to implement RFID installations consisting of smart shelves and RFID-tagged items to support in-store logistics and processes. In principle, they can also use these installations to implement planogram compliance verification: Each antenna is supposed to detect all tagged items in one location of the planogram. But due to physical constraints, RFID tags can be identified by more than one RFID antenna. Thus, one cannot decide if an item carrying such a tag complies with the planogram. We propose a new method called RPCV which checks planogram compliance on large databases of items. It is based on the observation that the number of times an antenna identifies each item of a certain product type roughly follows a normal distribution. RPCV represents each item as a two-dimensional vector containing the number of readings both by the right antenna and by wrong ones according to the planogram. It clusters this data, separately for each product type. A cluster then is a set of correctly placed items or of misplaced ones. RPCV produces one order of magnitude less wrong predictions than current state of the art, and it requires less data to yield good predictions. A study with RFID-equipped goods and smart shelves shows that our approach is effective in realistic scenarios.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications; H.3.3 [**Information Search and Retrieval**]: Clustering

General Terms

Algorithms, Experimentation, Performance, Reliability

Keywords

RFID, Planogram Compliance, Data Quality, Data Cleaning

EDBT 2010, March 22–26, 2010, Lausanne, Switzerland.

Figure 1: The smart shelf used in field trials

1. INTRODUCTION

Planograms are carefully designed layout plans that define the placement of products in a retail store. They specify which product should be placed at which location on which shelf in the store. Compliance with planograms is important, since optimal product placement can increase profit by up to 8.1% [5]. In principle, Radio Frequency Identification (RFID) [11] can be used to implement planogram compliance: The items in the retail store are equipped with RFID tags, the shelves with RFID readers that write a stream of RFID data into a database. One RFID reader controls many RFID antennas. Each antenna corresponds to one location defined in the planogram. Thus, one can verify planogram compliance by querying the database for the antenna identifying a certain item. Figure 1 shows such a setup, a shelf consisting of four shelf units. The antennas are the gray

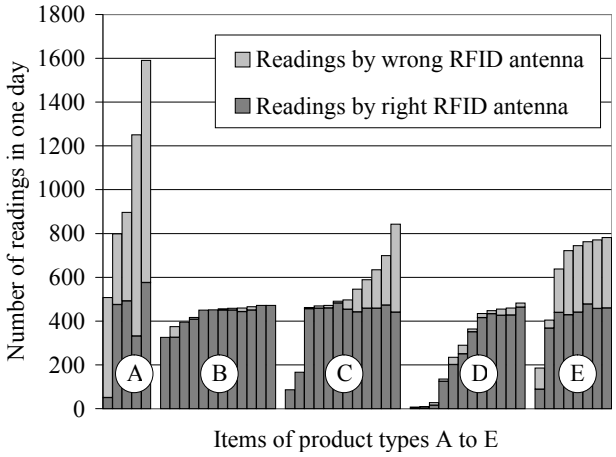

Figure 2: RFID data from field trials

boxes mounted on the bottom side of the shelf units. Each shelf unit has two antennas, i.e., the planogram can differentiate between the right and the left side of each shelf unit.

To ensure that all RFID tags are identified, retailers typically install many RFID antennas close to each other, resulting in dense RFID deployments with many RFID antennas near to each other. In such installations, RFID tags are frequently detected by more than one antenna. In this case, the system cannot decide where on a shelf a certain item is located. Furthermore, an RFID reader does not know the reading field of an RFID antenna, i.e., the area where it can detect RFID tags. This happens because objects in the vicinity of the antenna can reflect or absorb magnetic waves. These waves might cancel each other out, creating blind spots, or be superimposed and thus reach farther than expected [12]. This makes the integration of RFID data into business processes difficult, because reliable data is a prerequisite for many processes, like data mining of customer data to improve planograms, etc.

In this paper, we present the RFID Planogram Compliance Verification (RPCV) algorithm to check planogram compliance on large databases of RFID data. Our algorithm is motivated by the outcomes of field trials at a large German retailer. Figure 2 shows data of five product types collected during these field trials, which is representative. Each bar represents one item. Because the planogram of this field trial (not shown here) specifies a different number of items to be displayed on the shelf for each product type, the figure shows different numbers of items/bars for each product type. Subsequently, we refer to the antenna that should be identifying an item according to the planogram as the right antenna, to the other ones, e.g., antennas from neighboring locations, as wrong antennas. The dark gray bars show the number of times the right RFID antenna identifies an item. The corresponding light gray bar shows the stacked number of times an item was identified by wrong RFID antennas. For our approach, it is now important that a normal distribution describes quite accurately the number of times an antenna identifies each item of a certain product type. I.e., if x is 'the number of times the antenna identified an item', and $f(x)$ is 'the number of items identified x times', then $f(x) \sim N(\mu, \sigma^2)$. We have observed this effect in the field trials. It occurs because items of the same product type have roughly the same physical characteristics. We refer to the distribution of readings by the right antenna and the one by wrong antennas as the *reading pattern of the product type*. The data shows why it is difficult to decide which location an item is placed on: Although none of the items was misplaced or moved during the particular trial where we collected this data, each item was identified by wrong antennas at least once. Various items were identified more often by the wrong antennas than by the right antenna (Product A). Furthermore, the number of times an item is identified during a certain time interval varies a lot (e.g., Product D), because RFID antennas usually have "blind spots" where RFID tags cannot be easily identified. Thus, it is difficult to implement planogram compliance using RFID. Current solutions [15, 25] rely on filters, e.g., they assign each item to the antenna that has identified it more often. However, as Product A shows, these approaches may produce many wrong predictions with databases of real RFID readings.

RPCV defines a sliding window over the continuous data streams of the RFID reader, i.e., it considers the items identified in a given time interval. For each item, our method counts the numbers of readings in the sliding window. RPCV represents each item as a two-dimensional vector, where the first dimension is the count of the readings by the right antenna. The other dimension is the number of readings by wrong antennas. It then clusters this data, separately for each product type. The clusters will correspond to correctly placed items or to misplaced ones.

In general, the reading patterns of different product types will be different. This is because several parameters influence the reading pattern, e.g., the total number of items read by an RFID antenna and the proportion of misplaced items. In order to analyze our algorithm, we have carried out an extensive evaluation with a real installation (i.e., RFID-tagged items, smart shelves). To comply with the state of the art regarding user experiments/experiments with customer interactions, we let persons not involved in the design of our algorithm carry out the interactions. Our evaluation shows that RPCV is fast, scales up to large databases of items and RFID readings, returns one order of magnitude less wrong predictions and requires less data than current state of the art to produce good predictions.

Summing up, this paper makes the following contributions:

- We describe typical problems when implementing planogram compliance in retail, and we show how these problems can in principle be solved using RFID.
- We propose a new method called RPCV to determine if items identified by more than one RFID antenna comply with the planogram. RPCV transforms the data so that one can apply a clustering algorithm to decide on the location of RFID tags. To our knowledge, we are first to look at this problem as a case for clustering.
- Having implemented RPCV as a Java extension of a MaxDB database, we evaluate it by experiments with an RFID installation. Furthermore, we validate RPCV against various worst-case scenarios and perform scalability tests with artificial data sets of up to 10,000,000 RFID readings to confirm the efficiency of our method.

Paper outline: The next section describes our retail scenario and says how RFID can optimize planogram compli-

ance. Section 3 covers related work, and Section 4 presents our approach. We evaluate it in Section 5.

2. APPLICATION SCENARIO

In this section we describe the setting for planogram compliance in retail, which we use to motivate our work and to evaluate our method. With RFID-equipped items and smart shelves that track the items present in real time, a retailer can run applications like planogram compliance, i.e., ensuring that the placement of items in a retail store follows a specific layout plan. The potential is huge, since fast and accurate planogram compliance can increase profit by up to 8.1% [5]. E.g., applying such an increase to Walmart would yield 1.1 billion US$ more profit.

Planograms: A planogram specifies the location of products on a shelf unit. For each product type, it states the minimum number of items needed, e.g., 'The first row has to be filled.' It also specifies the arrangement of brands on each shelf, e.g., first all items of Brand A, then the ones of Brand B, as well as the arrangement of products, e.g., first Product 1 of Brand A, then the next one of this brand etc. In this paper, we restrict our attention to planograms specifying the number of items of a certain type per location. Planograms are important, because they improve the visual effect and the space productivity. They also optimize shelf space usage and reduce out-of-stocks. Attractive layouts increase impulse purchases, e.g., by placing complementary products like pasta and tomato sauce next to each other. Furthermore, customers tend to only buy items that are placed on the expected location, e.g., a customer would probably not buy a single package of pasta placed in-between cleaning materials.

Planogram Compliance: Planogram compliance is difficult to ensure. This is because planograms are complex and change frequently, because of new marketing campaigns or because of seasonal products. Frequently changing planograms make non-compliance difficult to detect. Another reason why this detection may be difficult are shelf units filled with items of other product types. For example, if 6 items have to be replenished, but 10 items are packaged in one box, a clerk might be inclined to put the remaining 4 items just anywhere on the shelf.

RFID Characteristics: RFID can be used to check planogram compliance. One RFID reader controls many RFID antennas (1:n relationship), i.e., the reader knows which antennas are identifying which items. We can check if a smart shelf holds the right items and if they are correctly placed on the shelf. However, in dense RFID deployments with many RFID antennas near to each other, many RFID antennas can identify the same item, and they frequently do. This occurs because of physical interference. In other words, the problem is a general one, and future RFID technology will not be able to avoid this effect. Thus, a system cannot decide on the location of such tags. Note that, from a data management perspective, there is no difference if an item is identified by antennas from the same RFID reader, or by antennas from different readers. In the context of planogram compliance, the following characteristics of RFID are important:

C1: Varying number of readings Because of physical interference, and because of items being sold and replenished, there is a broad variation in the number of times items are identified within a certain time interval [26].

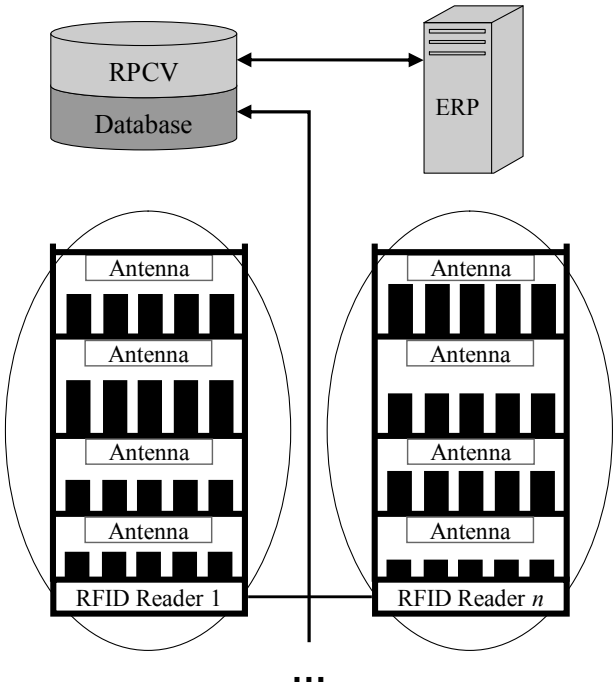

Figure 3: Sketch of components in our scenario

C2: Unpredictable reading field Because of physical effects like absorption and reflection, we cannot predict the reading field of an RFID antenna, i.e., the area where an antenna can identify RFID tags.

C3: Distinctive reading patterns The set of items of a product type follows a certain distribution. A normal distribution describes the number of readings of items of a certain type by the right RFID antenna well. The same holds for the number of readings by wrong antennas (cf. Figure 2).

C4: Large data volume RFID applications in retail are characterized by large numbers of items. This results in huge databases and might challenge the runtime performance of an estimator.

Planogram Compliance with RFID: We study the following general scenario:

- A retailer sells many items of different product types. All items are tagged with unique RFID labels.
- All shelves in the store are equipped with RFID readers, e.g., smart shelves [9]. Each RFID antenna is assigned to one location in the planogram, cf. Figure 3.
- Clerks replenish the shelves if they go empty, and they re-arrange items when planogram compliance is not given, or when the planogram changes.

Architecture of RPCV: Figure 3 is an overview of the components described in this scenario. RPCV is implemented as a database extension. I.e., one can query which items comply with the planogram in SQL using a database

procedure that implements RPCV. This makes integration into business applications easy, since one can adapt existing database queries to support RPCV without difficulty. Examples of business applications which might interact with RPCV are Enterprise Resource Planning (ERP) systems and Merchandise Information Systems (MIS).

3. RELATED WORK

While planogram compliance is well known to promote sales [1, 5], there is relatively little research on planogram compliance in smart shelves. Decker et al. [9] seem to be first using RFID for planogram compliance. In order to identify the correct location of an RFID tag on the shelf, their approach requires multiple RFID antennas per item, i.e., each smart shelf has many more antennas than items. Since there are millions of items in large retail stores, the approach is infeasible from an economic perspective. The database community has developed a number of approaches to predict the location of RFID tags identified by more than one RFID antenna, which we will discuss in the following.

Rule-based filters: Solutions from this category propose filtering data inside of a sliding window based on rules [2, 3, 7, 8, 13, 15, 20, 25], which can be directly applied to an RFID data stream or after the RFID data has been persisted. Examples of rules used are assigning the item to the first antenna which has identified it [2, 20], to the last antenna [25], i.e., assuming the most recent data is correct, and assigning the item to the antenna with the most readings [13, 15]. In the following, we call these methods FIRST, LAST and MOST, respectively. The methods are fast, but they can generate many wrong predictions, e.g., if we applied the method MOST to the data of Product A in Figure 2, three of the five items would be incorrectly classified as being misplaced, because wrong antennas identify these items more often than the right antenna. Since several commercial solutions implement such rules, e.g., IBM [20], SAP [7], and Siemens [2, 25], we will compare their accuracy and their performance to our method in Section 5.

Probabilistic filters: Methods based on probabilistic filtering are proposed in [14, 17, 18, 29]. These methods assign a certain probability of an RFID reading being correct to each reading, depending on predefined constraints and on statistical data. Examples of predefined constraints are that an item cannot be located on two shelves at the same time, and each shelf only stores a certain number of items. Statistical data consists of the probabilities of certain events, which are devised from history data or from samples gathered manually, like the probability that RFID fields of two RFID antennas overlap. However, such data is often specific to the position of the shelves in the store and to the items placed in each shelf. This is because objects in the vicinity of the RFID reader might reflect or absorb magnetic waves, and because items of different product types and shelves with different layouts have different physical characteristics. Since RFID installations might be very large, and layouts of shelves in retail change very frequently, it is often not practical to take samples to compute such statistical data. Thus, it is difficult to indeed deploy such methods in retail scenarios. Note that such methods could profit from RPCV, since our method could provide new statistical data as input. It could provide the probability that a specific item is misplaced or correctly placed, based on the reading pattern of its product type.

Particle filters: [21, 22, 27] use particle filters to infer the location of RFID tags. Particle filters represent a probability distribution of events through generated samples (particles). The samples are updated over time and are used to compute estimations. Such methods yield good results when RFID tags (or RFID readers) are in movement. This is because samples are updated when an RFID tag moves past several RFID antennas. In the retail scenario, duplicate readings are not caused by moving items, but by overlapping RFID reading fields. An item that is frequently identified by a wrong RFID antenna nearby would generate many particles at the wrong location. The particle filter would yield bad results.

Other methods: [23, 24] have studied algorithms optimized for RFID tags in movement. Different algorithms are presented that assign an RFID tag to a certain location after it has been identified many times or by more than one RFID antenna along the supply chain. Again, we think that these methods do not apply to our scenario. [6] proposes attaching several RFID tags to each item in different orientations. In such a setup, there is a very high probability that at least one RFID tag does not lie inside a "blind spot". Therefore, the variation in the number of readings (cf. Characteristic C1) might be small. But this does not help verifying planogram compliance, since an item with many RFID tags will be identified by the same number of antennas or even more than an item with only one tag.

Summing up, only the methods FIRST, LAST and MOST can be directly used to verify planogram compliance. Therefore, we will compare RPCV to these methods in Section 5.

4. RPCV ALGORITHM

We now present RPCV, a method to determine which RFID tags identified by more than one RFID antenna comply with the planogram. RPCV is fast, scalable, and it returns one order of magnitude less wrong predictions than related work. Furthermore, it requires less data than current state of the art to produce good predictions, i.e., the size of the sliding window needed until a certain accuracy is reached is small.

RPCV is motivated by the following observation from the field trials: The number of times an antenna identifies the items of a certain product type roughly follows a normal distribution (cf. RFID characteristic C3). In a nutshell, RPCV works as follows: It first counts the number of readings per RFID antenna for each item. As observed in the field trials, RFID readers identify item sets at different rates (cf. RFID characteristic C1), thus the number of readings per item may vary broadly. To reduce this variation, RPCV applies a low-pass filter. After that, it applies a clustering algorithm to all items of a certain product type. Each item is represented by a vector consisting of the number of readings by the right RFID antenna and the number of readings by wrong antennas. RPCV then decides for each cluster if it is misplaced.

Note that RPCV can be easily extended to support product types placed on more than one location, e.g., by modeling them as different logical product types, or by considering a set of right antennas instead of just one.

4.1 RPCV Algorithm in Detail

To describe RPCV in more detail, we mention that it requires two data structures:

- $RfidReads$: a multiset with tuples (i, ant) of items i and the RFID antennas ant identifying these items within a pre-determined time interval (sliding window).
- $Planogram$: a map whose keys are product types t, and the value corresponding to t is the RFID antenna ant that should be identifying items of type t.

The sliding window determines the body of RFID data available to RPCV. The scenario determines the size of the sliding window, i.e., by means of the time intervals a retailer rearranges misplaced items, and how often the RFID readers are polled. We analyze the effect of the sliding window on the quality and on the runtime of RPCV in Section 5.

```
1: input RfidReads, Planogram
2: MisplacedItems = {}
3: for all (productType t ∈ Planogram.getKeys()) do
4:    P = {}
5:    ant = Planogram.get(t)
6:    for all (item i in RfidReads) do
7:       if (item i is of type t) then
8:          readsR = |{(i,a) ∈ RfidReads : a = ant}|
9:          readsW = |{(i,a) ∈ RfidReads : a ≠ ant}|
10:         P = P ∪ {(i, readsR, readsW)}
11:      end if
12:   end for
13:   P* = all pairs (readsR, readsW) in P
14:   P* = lowpassFilter(P*)
15:   {P1, P2} = Cluster(P*, 2)
16:   for all (cluster Pj) do
17:      if avg(Pj.readsW) > avg(Pj.readsR) then
18:         insert the elements of P corresponding to the vectors contained in Pj into MisplacedItems
19:      end if
20:   end for
21: end for
22: output MisplacedItems
```

Algorithm 1: RPCV Algorithm

Algorithm 1 describes RPCV. It iterates through all product types (Line 3). First, it determines the RFID antenna ant that should be identifying each product type according to $Planogram$ (Line 5). For each item, RPCV counts the number of readings by the right antenna ($readsR$) and the number of readings by other (wrong) antennas ($readsW$) and adds this two-dimensional vector to a set P corresponding to type t (Lines 6-10). In Line 13, RPCV removes attribute i from P, since it is not used for clustering. RPCV then applies a low-pass filter to reduce the variation of the remaining values $readsR$ and $readsW$ (Line 14). Now it applies a clustering algorithm to the values $readsR$ and $readsW$ of each item, to create two clusters P_1 and P_2 for each product type (Line 15). We will explain later why the number of clusters to be identified is two. The first argument of function $Cluster$ in Line 15 is the set of vectors to be clustered, the second argument is the number of clusters to be returned. Then RPCV decides if a cluster represents misplaced items. It does so by checking if on average the cluster was identified more often by wrong antennas (Lines 16-20). After RPCV has iterated through all product types, it returns the set of misplaced items. The choice of the clustering algorithm and the low-pass filter are explained in the following subsections.

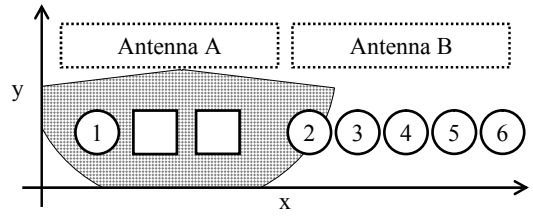

Figure 4: Example scenario

Example 1 introduces a simple scenario for planogram compliance. We will use this scenario in subsequent examples to illustrate RPCV and to illustrate the clustering algorithm and the low-pass filter.

Example 1: Figure 4 shows two RFID antennas and items of two different product types, located in a 2D-space. The product types are represented as circles and squares. The gray area in the figure shows the reading field of Antenna A. We now want to decide if the Items 1 to 6, which are of product type circle, comply with the planogram. Suppose that the planogram requires these items to be placed under Antenna B. Item 1 is identified very often by Antenna A. Item 2 is identified often both by Antennas A and B, and Items 3 to 6 are identified very often by Antenna B.

4.2 Clustering: Expectation Maximization

Items of the same product type have similar physical characteristics, i.e., they tend to affect the magnetic waves of the RFID reader in a similar way. Thus, items of the same type might be read at wrong locations in the shelf with a similar probability. RPCV takes advantage of this by incorporating a clustering algorithm. It is applied to the value pairs $readsR$ and $readsW$ of each item.

In a set of preliminary experiments, we have evaluated different clustering algorithms [28]. We have integrated these clustering algorithms into RPCV and have counted the number of false predictions with the data from the field trials. Our results were as follows: density based clustering (11% wrong predictions), farthest first traversal algorithm (10% wrong predictions), k-means (9% wrong predictions), and Expectation Maximization (1% wrong predictions). In other words, the Expectation Maximization (EM) algorithm [10] provides the best results for our use-case.

```
1: input tuples {(x,y)}, number of clusters k
2: {P1..., Pk} = initClusters({(x,y)}, k)
3: {θ1..., θk} = initDistributions({P1, ...Pk})
4: while (clusters change) ∧ (# of iterations < limit) do
5:    for all (cluster Pj) do
6:       Ej = estimateExpectedValues(Pj, θj) // E-Step
7:       θj = estimateParameters(Pj, Ej) // M-Step
8:       Pj = updateCluster(Pj, θj)
9:    end for
10: end while
11: output {P1, ..., Pk}
```

Algorithm 2: EM Algorithm

In the next section we experiment with setups where RPCV works well, and setups where it does not. To understand the experiments well, one should know how the EM algorithm

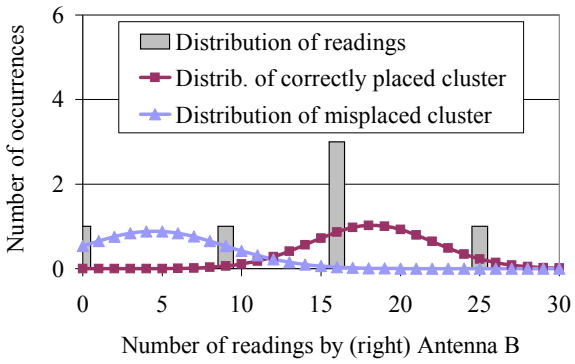

Figure 5: Distribution of raw readings

works. This is why we list it in Algorithm 2. The EM algorithm tries to fit k probability distributions to each dimension of the input data. The result will be k clusters, and one probability distribution in each dimension describes a cluster. Thus, since RPCV has two-dimensional input data, two probability distributions will represent one cluster. The algorithm starts with an initial set of clusters on the two-dimensional data $\{(x, y)\}$ (Line 2), which is gained by running a few iterations of the k-means algorithm. The EM algorithm estimates the parameters of the distributions of the initial clusters (Line 3) and improves them iteratively (Lines 4-10). It iterates until the clusters do not change any more, or until a limit of 100 iterations is reached. 100 iterations are in line with best practices described in literature [28]. For each cluster and in each iteration, it first computes the expected values of the log likelihood with respect to the current distributions (Expectation-Step, Line 6) and then updates the parameters of the distributions using the expected values computed in the E-step (Maximization-Step, Line 7). At the end, the algorithm outputs k clusters. As the probability distribution we use the normal one. It correctly models the RFID data, and it can be computed fast, since a closed form of the estimators exists [10]. Example 2 illustrates how RPCV uses the clustering algorithm.

Example 2: In this example, which continues Example 1, we look at the readings by Antenna B. Within the sliding window, Antenna B identifies each item with the following frequency: Item 1: 0 times, Item 2: 9 times, Items 3 to 5: 16 times, Item 6: 25 times. First, RPCV will group these items by the number of readings, shown in Figure 5. The bars show the number of occurrences of each number of readings. The figure also shows the two normal distributions that the EM algorithm fits to the data. Each curve represents one cluster. RPCV decides if a cluster represents misplaced items by checking if on average the items in the cluster were identified more often by wrong antennas. Each bar is assigned to the most likely normal distribution, i.e., to the curve with the highest value at the x-position of the bar. The bars with the frequencies 0 and 9 belong to one cluster, which turns out to be a cluster of misplaced items. The other two bars belong to the other cluster, which turns out to be a cluster of correctly placed items.

Internally, the EM algorithm uses probabilities to express the membership of items in clusters. In our scenario, this fuzzy boundary yields better results than fixed memberships: During the iterations of the clustering algorithm, there is a higher chance that the assignment of items to a certain cluster changes, thus reducing the chance of the clustering algorithm getting stuck at local optima. For instance, consider Figure 5 in Example 2: The item which was identified 9 times has a high likelihood of belonging to the misplaced cluster and a low likelihood of belonging to the other one.

As mentioned, we configure the EM algorithm to find two clusters, because there are two kinds of items we want to separate: correctly placed and misplaced items. We have experimented with different numbers of clusters, and two clusters have lead to the best results. Looking at the raw RFID readings, there are items that are definitely correctly placed or misplaced, and there are items whose actual location is difficult to predict, e.g., because two antennas identify the item very often. With more than two clusters, each of these items might form its own cluster. Then we cannot decide on the location of these "difficult" items. With two clusters, we likely force the EM algorithm to assign such items either to the cluster of correctly placed items or to the one of misplaced items. Note that, in general, both clusters can represent correctly placed or misplaced items, e.g., if none of the items in the cluster or all of them are misplaced. To cope with such situations, RPCV decides if a cluster represents misplaced items by checking if on average the cluster was identified more often by wrong antennas.

4.3 Low-Pass Filter

Even though the number of times an antenna identifies the items of a certain product type roughly follows a normal distribution, the absolute numbers of RFID readings of individual items might differ a lot. This occurs due to blind spots, due to customers buying items, and due to items being replenished (cf. RFID characteristic C1). This poses a problem to clustering algorithms: As we have observed in the field trials, there are items that are identified only once in a certain time period, while others are identified hundreds of times.

To overcome this problem, RPCV applies a low-pass filter to the RFID data. Low-pass filters let numbers with small values pass through, and they reduce the value of large numbers. A low-pass filter for our scenario must fulfill the following requirements: (1) It must significantly reduce the variation of the numbers of readings, and (2) it must support a large number of items (cf. Characteristic C4). Because of (2), only simple filters are applicable in our scenario. Many mathematical functions can be computed quickly and fulfill Requirement (1). As a first step, we have identified such functions, e.g., roots, logarithms, and sigmoid functions. Next, in another set of preliminary experiments, we have evaluated these functions with different parameters and have applied them to the data from the field trials, in combination with the clustering algorithm. We did obtain the best results using the square root as a filter, since it has produced the smallest number of false predictions. The square root will reduce large values by much, i.e., from items with a large number of readings. Further, it imposes little change on small values, thus reducing the variation of the number of readings. We show function $lowpassFilter$ in Equation 1.

$$lowpassFilter(\{(x, y)\}) = \{(\sqrt{x}, \sqrt{y})\} \quad (1)$$

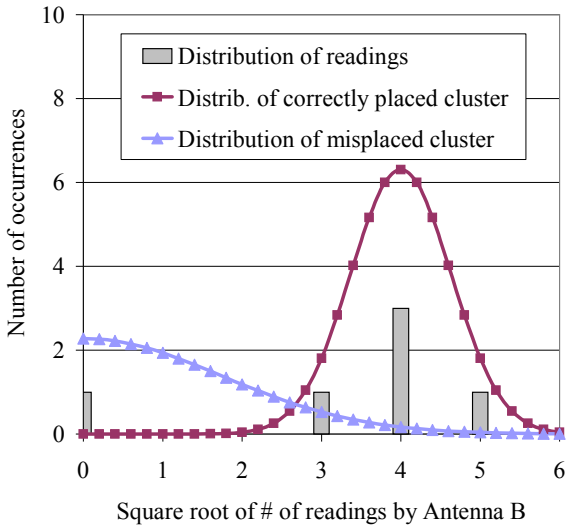

Figure 6: Distribution of filtered readings

Example 3 illustrates the low-pass filter. It shows the impact of the filter on the clusters determined by the EM algorithm.

Example 3: In Figure 5 there is a big difference between the number of readings for each item, and the bars have a similar distance to each other in the x-direction. The EM algorithm has problems correctly classifying such data, and the column at frequency 9 (Item 2) is incorrectly assigned to the cluster of misplaced items. Figure 6 shows the results when we apply a low-pass filter to the number of readings. Now the data is distributed over a smaller interval, and large values lie closer to each other. The EM algorithm classifies the filtered data correctly.

5. EVALUATION

In this section we evaluate RPCV. We want to (1) provide an intuition of how RPCV works, (2) show that the accuracy of RPCV is better than the one of related work, (3) show that RPCV requires less data to produce good predictions, (4) identify scenarios where RPCV does not function well, and (5) show that RPCV is fast when predicting with a large number of items and a large number of RFID readings. We compare RPCV with the methods FIRST, LAST and MOST, since they are the only methods from related work that can be directly used to verify planogram compliance. These methods assign each item to the first RFID antenna identifying an item, to the last one, and to the one with the most readings, respectively.

We have implemented our method as an extension of SQL in the SAP MaxDB database management system. We ran all experiments on a desktop PC (Windows, 2GB RAM, 2GHz dual core CPU, 1 SATA hard disk). We used the default configuration of MaxDB [4] and Java (version 1.6.0_12) for the performance experiments.

5.1 Experimental Setup

We evaluate our approach with the scenario described in Section 2. We have used real-world data obtained from the smart shelf shown in Figure 1, and we also experiment with synthetic data.

Real-World Data

We have equipped a retail shelf with two RFID readers (Intermec IF5). The shelf comes from a large German retailer and is identical to the one used in their stores. Its dimensions are 166cm x 110cm x 65cm. The shelf consists of four shelf units, and each shelf unit contains two RFID antennas, c.f. Figure 1. The shelf contains items belonging to 5 different product types. For this experimental setup, we identify the product type whose location prediction is most difficult. This is the product type with the worst reading pattern, i.e., the right RFID antennas cannot always identify items of this product type, and antennas from nearby locations often generate readings of these items.

On average, a retailer has between 8 and 13 items of each product type on one shelf [19]. For our experiments, we choose a challenging scenario with many more items than average. We used 30 items of the product type with the worst reading patterns, and we did experiments in three kinds of scenarios that occur in a retail store:

1. **Static scenario:** We arrange the items on the shelf in different ways, with misplaced items ranging between 0% and 50%. After the items were rearranged, we start recording the RFID data.

2. **Sales scenario:** We arrange the items on the shelf in the same fashion as in the static scenario. After we started recording the RFID data, different items which were not misplaced are removed from the shelf one by one.

3. **Replenishment scenario:** The experimental setup is the same as in the sales scenario, but after all items were removed, the shelf is replenished with new items.

In the static scenario there are no customer interactions like sales or replenishment. To ensure that we do not bias the customer interactions in the other two scenarios, we have asked a person unfamiliar with RPCV to remove and replenish items. This is in line with state-of-the-art user experiments, i.e., the individuals who have designed the experiment must not be part of it. The two scenarios with customer interactions are challenging: In the sales scenario, some items identified by the right antenna will have a large variation in the number of readings. In the replenishment scenario, all correctly placed items are replaced.

We poll the RFID readers three times per minute for 10 minutes. One polling corresponds to one RFID reading cycle. Thus, our sliding window consists of 30 reading cycles. Our smart shelf produced 34,851 RFID readings. In total, we have predicted the location of items 299 times.

Synthetic Data

We also experiment with synthetic data. This allows us to independently vary each parameter that might influence RPCV, to repeat tests more often, and to test extreme parameter values.

There are many uncertain parameters influencing the number of RFID readings, and we cannot determine the number of readings analytically. Therefore we generate different RFID reading patterns by means of a simulation. Our simulation is based on Monte Carlo Methods [16], and it works

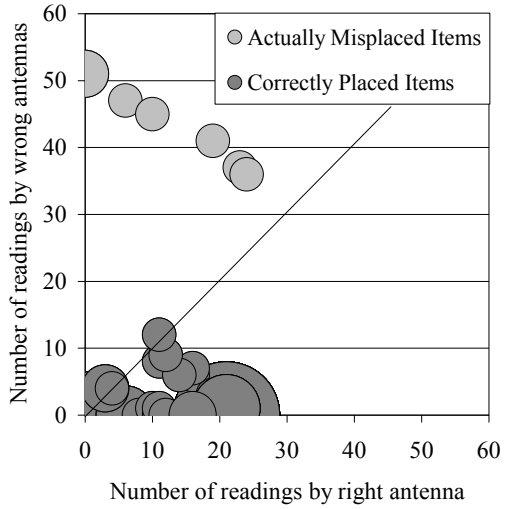

Figure 7: Replenishment scenario, 25% misplaced

as follows: We simulate a reading by each antenna in a time interval, and we do so repeatedly: For each time interval, we decide with a fixed probability p if the right antenna identifies an item, and with a fixed probability q if a nearby antenna does so. If an antenna identifies an item, we determine the number of times the item was identified by drawing a number from a normal distribution $N(\mu = 0.50; \sigma^2 = 0.25)$, multiplying it with 100 and converting it to an integer. This simulation is fast, and it creates data very similar to the data from the experiments with the smart shelf.

Based on observations from the real-world data, we simulate three kinds of RFID reading patterns: (1) correctly placed items that are frequently identified by the right antenna and rarely by wrong antennas, (2) correctly placed items that are frequently identified by the right and by wrong antennas, and (3) misplaced items that are frequently identified by wrong antennas and rarely by the right antenna. Note that our simulation draws the number of readings from a probability distribution, therefore reading pattern (2) will also contain correctly placed items which are identified more often by wrong antennas than by the right one.

5.2 Evaluation with Real-World Data

Intuition

In order to provide an intuition regarding RPCV, we have conducted an experiment with real data where 25% of all items are misplaced. Figure 7 shows the results. The x-axis shows the number of RFID readings by the right RFID antenna, and the y-axis the number of RFID readings by wrong RFID antennas. The dark gray circles represent items correctly placed, and light gray circles represent actually misplaced items. The size of the circles represents the number of items at that position in the figure, e.g., the light gray circle most to the right represents one item, and the light gray circle most to the left represents two items. Items above the diagonal were identified more often by wrong antennas than by the right one. For instance, the light gray circle that is most to the right represents one item that was identified 24 times by the right antenna and 36 times by wrong ones.

Table 1: Accuracy

Method	*Precision*	*Recall*	F_1 *score*
RPCV	98.2%	96.4%	97.3%
MOST	63.4%	100.0%	77.8%
FIRST	56.0%	100.0%	71.8%
LAST	38.5%	92.9%	54.5%

The items identified by the right antenna show a small number of RFID readings, on average around 12 readings, because of constantly changing items resulting from sales and replenishment. The misplaced items are identified very often, since they are not sold. On average they are identified around 44 times at their actual location.

RPCV identifies one cluster of misplaced items. Items in this cluster have an average of around 44 readings by wrong antennas, and around 8 readings by the right antenna. Further, RPCV identifies one cluster of correctly placed items that show less than one reading by wrong antennas on average, and around 12 readings by the right antenna.

In this experiment, the data is clearly segregated, and RPCV produces no wrong prediction. MOST in turn has difficulties in this experiment: 8 correctly placed items were identified equally or more often by wrong antennas. Thus, MOST results in 8 wrong predictions. The methods FIRST and LAST will produce 8 and 3 wrong predictions, respectively.

Accuracy with Real-World Data

In this set of experiments we compare the accuracy of RPCV to the one of related work on the data gathered with our real-world installation.

We measure accuracy with the commonly used measures precision, recall and the F_1 score. They are calculated using the number of true positives (TP), false positives (FP) and false negatives (FN). True positives are misplaced items predicted as such, false positives are correctly placed items with incorrect prediction, and false negatives are misplaced items, but the prediction is that the placement is correct. Precision is defined as $\frac{TP}{TP+FP}$, recall is defined as $\frac{TP}{TP+FN}$. The F_1 score is defined as $\frac{2 \cdot precision \cdot recall}{precision+recall}$.

Table 1 lists the accuracies. From 299 predictions, RPCV was wrong in only 3 cases. We will further examine these wrong predictions in the next set of experiments. This results in an F_1 score of 97%. The next best related work – method MOST – produces 32 wrong predictions, i.e., an F_1 score of 78%. Thus, our method produces one order of magnitude less wrong predictions than other approaches.

Incorrect Predictions with Real-World Data

Now we want to identify the situations when RPCV does not perform well with real-world data. Figure 8 shows one experiment from the static scenario, where 50% of the items are misplaced. Predictions with data from this experiment are difficult, since three correctly placed items (shown in two circles) were frequently identified by wrong antennas. Therefore, in the figure these two items are very close to the items actually misplaced. When applying our algorithm, one correctly placed item is predicted to be misplaced (false positive). In the figure, a dashed rectangle encloses this item. MOST, by the way, will produce two false positives as well in this specific setting. Figure 9 shows an experiment from

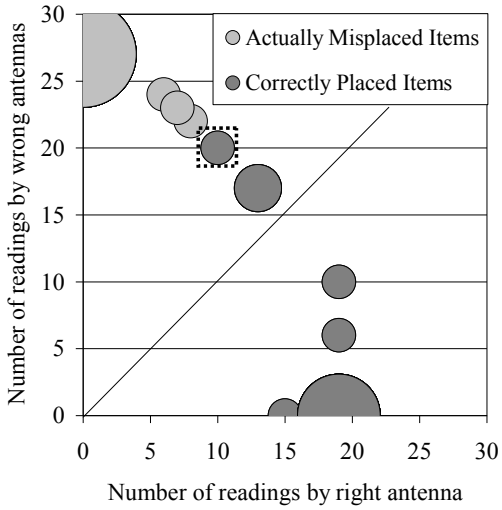

Figure 8: Static scenario, 50% misplaced

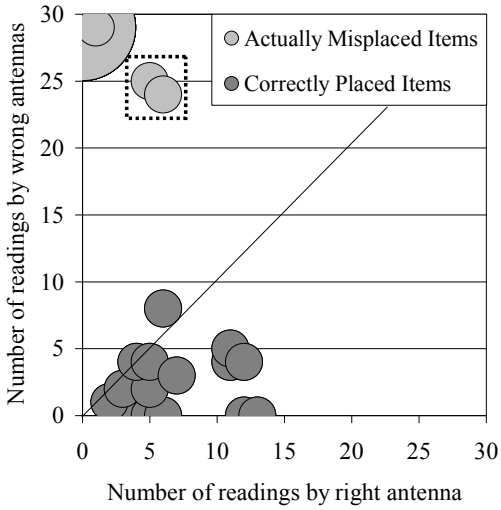

Figure 9: Sales scenario, 50% misplaced

the sales scenario, where 50% of the items are misplaced. In this experiment, RPCV produces two false negatives. A dashed rectangle encloses both items in the figure. MOST will produce two false positives in this setting.

The wrong predictions in both scenarios have to do with the way the EM algorithm works. It tries to fit two normal distributions to each dimension of the input data, and one distribution of each dimension will represent one cluster. We illustrate this with the distribution of readings by wrong antennas, shown in Figure 10. The x-axis shows the number of readings by wrong antennas, and the y-axis shows the respective number of occurrences. The bars show the distribution of readings of the RFID data, i.e., the bar most to the right is equivalent to the biggest circle on the top left of Figure 9. Each curve shows the normal distribution for one cluster. RPCV obtains these curves by fitting two normal distributions to the data in the figure. The EM algorithm assigns each item with a certain number of readings to the most likely normal distribution, i.e., each bar is assigned to the curve with the highest value directly above the bar. Note that the curves are skewed because RPCV fits them to the square root of the data, because of the low-pass filter. A dashed rectangle encloses the two wrong predictions from Figure 9. The normal distribution of the cluster of misplaced items has a higher mean value of the number of readings by wrong antennas than the other normal distribution. It has a very high mean value and a very small standard deviation. The two wrong predictions do not lie under this curve. The EM algorithm chooses the distributions with the highest likelihood, and this results in wrong predictions.

We only encountered this kind of problem in the experiments described in this section. Furthermore, we did not find a correlation between this problem and the position of the item on the shelf. In other words, this effect seems to occur very rarely. Nevertheless, we plan to explore the issue further in future work, e.g., by combining the results of the EM algorithm with clustering algorithms that do not show this behavior.

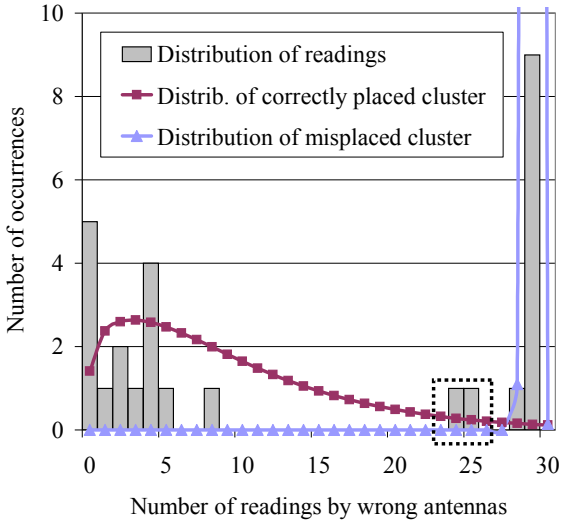

Figure 10: Distribution of readings by wrong antenna

5.3 Evaluation with Synthetic Data

We now want to study the influence of the various parameters of RPCV. We do so by means of simulations. They let us vary parameters independently, repeat tests more often, and test extreme parameter values. In particular, we vary the number of RFID readings, the total number of items, the number of items for each kind of reading pattern, and the probabilities of items being identified by the right RFID antenna and by wrong ones.

Size of Sliding Window

In this set of experiments, we want to analyze the effect of the size of the sliding window on the accuracies of RPCV and of related work. We set the parameters of our simulation to mimic characteristics of the real-world data, as studied before. We simulate the static scenario with 30 items, with half of the items correctly placed and identified very often by the right antenna, one fourth of the items correctly placed

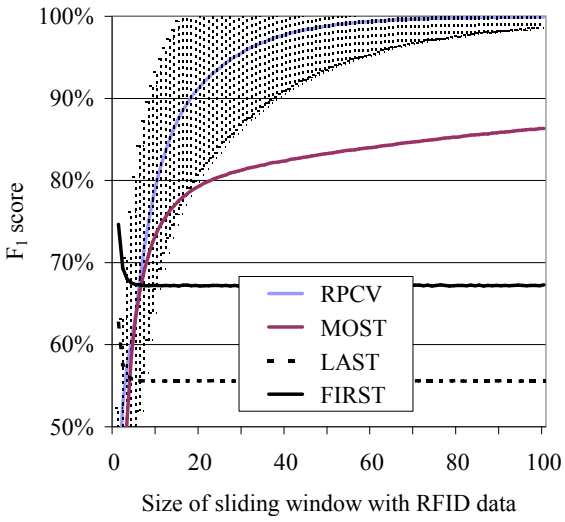

Figure 11: F_1 scores for different sizes of the sliding window

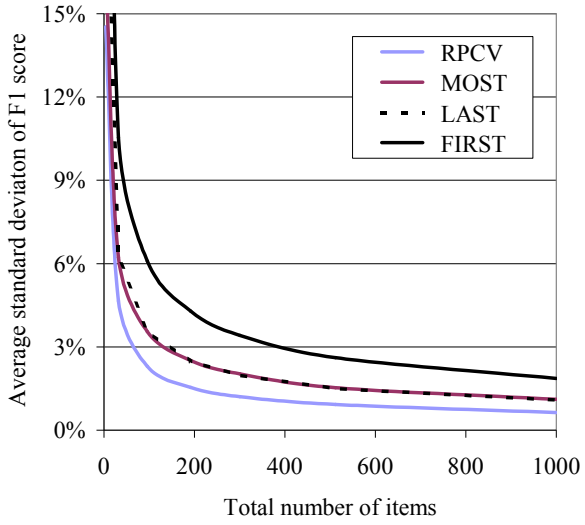

Figure 12: Standard deviation vs. number of items

and identified very often by the right and by wrong antennas, and with one fourth of the items being misplaced and identified very often by wrong antennas. We compute predictions after the first RFID reading cycle, after the second one etc. until reaching 100 cycles. We compare the values of the F_1 scores.

We expect the F_1 score to be constant with methods FIRST and LAST, since we are simulating a static scenario. The F_1 score of RPCV and of MOST should stabilize, because the share of readings by each antenna should become more stable as more RFID data is gathered.

We show the average results of 1,000 experiments in Figure 11. As expected, the F_1 score of FIRST and LAST is constant. The F_1 score of RPCV and of MOST increases slightly with the size of the sliding window. The F_1 scores with a sliding window of 30 are identical to the ones of our real-world experiments. They are similar to the F_1 scores shown in Table 1. The dotted area in the figure plots the standard deviation of RPCV. After 18 reading cycles, the average result of our method plus the standard deviation is still better than the second best method. The standard deviations of the other methods are not plotted for better readability. The standard deviation of MOST is nearly constant at 7 percentage points. The one of FIRST is constant around 11 percentage points, the one of LAST is constant at around 6 percentage points.

These experiments show that RPCV produces good results with a small sliding window, and that the prediction accuracy, i.e., the F_1 score, quickly reaches 100%. Thus, our method is applicable in scenarios that need to track planogram compliance in small time intervals.

Total Number of Items Present

We now want to find out how the total number of items present influences our prediction, and if there is a minimum number of RFID items that has to be present for RPCV to work.

Our simulations have the same parameters as before. Recall that a retailer has between 8 and 13 items of each product type on a shelf on average. We vary the total number of items present between 4 and 1,000. A shelf in retail holds less than 1,000 items – this number is a worst case. For each number of items we run 1,000 tests for each sliding window size from 1 to 100 reading cycles.

We expect the average results of RPCV to be independent from the total number of items, and we expect the standard deviation to increase as the total number of items decreases, since RPCV will have less data to carry out its predictions.

Our expectation holds: The average results of RPCV do not vary with the total number of items. To analyze the standard deviation, we calculate the average deviation of all tests for each total number of items. The results are shown in Figure 12. The average standard deviation lies around 15% with 4 items, and it decreases as the total number of items increases. The average standard deviation of RPCV is lower than the ones of the related approaches. Since the average results of RPCV do not vary with the total number of items, it is applicable for product types with very few items.

Worst-Case: Proportion of Misplaced Items and Probabilities of Items Being Identified

In this set of experiments we vary the number of items for each kind of reading pattern while keeping the total number of items constant (100). Further, we vary the probabilities of items being identified by each RFID antenna independently, i.e., we vary the values of p and q between 0 and 1. We want to find the setting which is most challenging for our algorithm.

When varying the number of items for each reading pattern we found the worst results when the number of items identified frequently by the right antenna and by wrong ones is very high. The overall probabilities of items being identified have little impact on the average quality of RPCV, but they will increase the standard deviation. Reducing the overall probabilities will result in less data, similarly to experiments with a smaller sliding window.

Next, we vary the probabilities from different RFID antennas independently, i.e., p and q. We found that the main factor influencing RPCV is the difference between the probabilities of items being identified by each antenna. The results

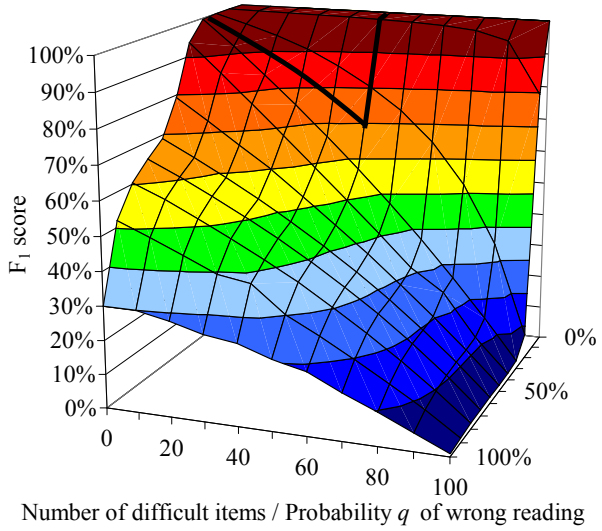

Figure 13: F_1 score for worst-case scenarios

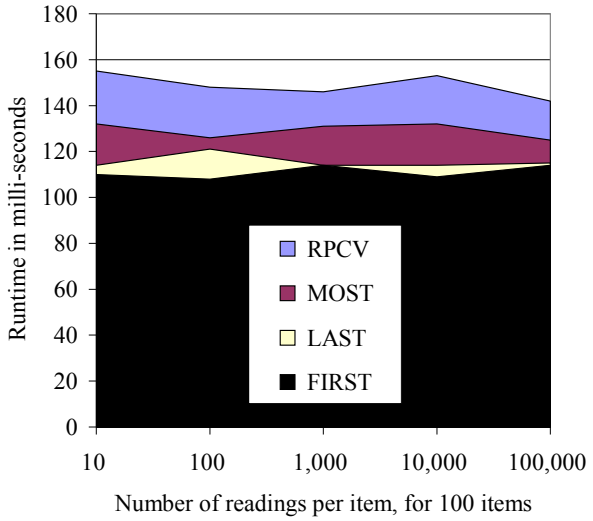

Figure 14: Runtime in dependence of the # of readings

are presented in Figure 13. The graph shows the average result of 1,000 tests with 100 items and 100 reading cycles. The x-axis shows the number of items identified frequently by the right antenna and by wrong ones, denoted as "difficult items". For these items, we fix the probability of an item being identified by the right antenna to $p = 50\%$, and we vary the probability q of being identified by wrong antennas (y-axis). The remaining number of items is equally divided amongst items correctly placed and identified frequently by the right antenna and misplaced items identified frequently by wrong antennas.

In general, the F_1 score of our method lies above 70% when less than half of all items are identified very often by the right antenna and by wrong ones, and when the probability q that items are identified by wrong antennas is half of p, i.e., 25% in the graph. The thick lines in the graph show this interval. Note that this interval covers the usual reading characteristics of RFID readers. The parameters under which our method does not function well correspond to reading characteristics that are unrealistic.

Performance and Scalability

Now we evaluate the performance and scalability of RPCV. We analyze the runtime of RPCV and compare it to the one of related work. In this experiment we use several databases with different sizes. Recall that a retailer has between 8 and 13 items of each product type on one shelf on average. We evaluate the performance with 100 items. The number of times each item is identified is varied between 10 and 100,000. Thus the largest database table in this experiment has 10,000,000 rows. Figure 14 graphs the results. Each number is the average of 20 runs. The runtime of RPCV lies between 140ms and 160ms. Even though it is slower than related work, it still is fast for a large number of items and a huge number of RFID readings. Further, RPCV requires less data to produce good predictions and generates one order of magnitude less wrong predictions than current state of the art.

In the next experiment we keep the total number of RFID readings for each database constant at 10,000,000 readings and vary the number of items between 100 and 1,000 items. Even though it is very unlikely for a retailer to have more than 100 items of a product type on shelf, we still want to evaluate how RPCV scales in extreme situations. See Figure 15. The runtimes of FIRST and LAST lie around 100ms, and the growing number of items does not affect them. For 100 RFID readings, the runtime of MOST is 125ms, and it grows nearly linearly to 182ms for 1,000 readings. RPCV grows nearly linearly as well: 124ms for 100 readings, and 291ms for 1,000 readings. The fact that the growth is almost linear is backed by more data which is omitted from the graph for readability.

6. CONCLUSIONS

RFID-based tagging to optimize commodity flows is important in all industry segments. However, not being able to decide on the actual location of RFID tags identified by more than one RFID antenna poses problems when integrating RFID data with enterprise-backend systems. This occurs because of physical interferences. This phenomenon is principal in nature, and future RFID technology will face it as well.

In this paper we have presented the RFID Planogram Compliance Verification (RPCV) method to decide if RFID tags identified by more than one RFID antenna comply with the planogram. RPCV is motivated by the observation that the number of times an antenna identifies each item of a certain product type roughly follows a normal distribution. It represents each item in the database as a two-dimensional vector containing the number of readings both by the right antenna and by wrong ones for a given time interval. RPCV clusters this data, separately for each product type. It then deems all items in a cluster correctly placed or misplaced. RPCV requires less data to produce good predictions and produces one order of magnitude less wrong predictions than related work. It is fast and scalable. An analysis and extensive experiments both with synthetic databases of items and with a real RFID installation confirm the applicability of our approach both in extreme settings and in realistic scenarios.

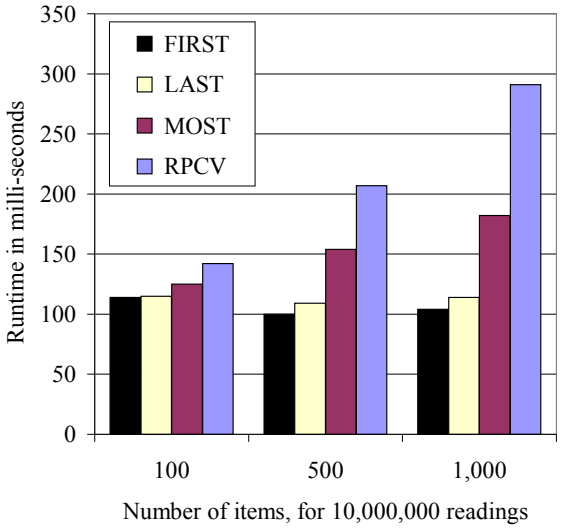

Figure 15: Runtime in dependence of the # of items

Acknowledgements

The work presented in this paper was partly funded by the German government (Bundesministerium für Bildung und Forschung) through the projects LoCostix and Polytos.

7. REFERENCES

[1] R. Bai and G. Kendall. A model for fresh produce shelf-space allocation and inventory management with freshness-condition-dependent demand. *Informs Journal on Computing*, 2008.

[2] Y. Bai, F. Wang, and P. Liu. Efficiently filtering RFID data streams. In *Proceedings of CleanDB Workshop*, 2006.

[3] Y. Bai, F. Wang, P. Liu, C. Zaniolo, and S. Liu. RFID Data Processing with a Data Stream Query Language. In *Proceedings of ICDE'07*, 2007.

[4] A. Bögelsack, S. Gradl, M. Mayer, and H. Krcmar. *SAP MaxDB Administration*. SAP Press, Harlow, 2008.

[5] W. Bishop. Documenting the value of merchandising. Technical report, National Association for Retail Merchandising Service, 2000.

[6] L. Bolotnyy and G. Robins. The Case for Multi-Tag RFID Systems. In *Proceedings of WASA'07*, 2007.

[7] C. Bornhövd, T. Lin, S. Haller, and J. Schaper. Integrating automatic data acquisition with business processes: Experiences with SAP's Auto-ID Infrastructure. In *Proceedings of VLDB'04*, 2004.

[8] J. Brusey, C. Floerkemeier, M. Harrison, and M. Fletcher. Reasoning about uncertainty in location identification with RFID. In *Proceedings of RUR Workshop*, 2003.

[9] C. Decker, U. Kubach, and M. Beigl. Revealing the Retail Black Box by Interaction Sensing. In *Proceedings of ICDCSW'03*, 2003.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society, Series B 39*, 1977.

[11] K. Finkenzeller. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley & Sons, Inc., 2003.

[12] C. Floerkemeier and M. Lampe. Issues with RFID Usage in Ubiquitous Computing Applications. In *Proceedings of Pervasive'04*, 2004.

[13] M. J. Franklin, S. R. Jeffery, S. Krishnamurthy, and F. Reiss. Design Considerations for High Fan-in Systems: The HiFi Approach. In *Proceedings of CIDR'05*, 2005.

[14] S. R. Jeffery, M. J. Franklin, and M. Garofalakis. An adaptive RFID middleware for supporting metaphysical data independence. *The VLDB Journal*, 2008.

[15] S. R. Jeffery, M. Garofalakis, and M. J. Franklin. Adaptive cleaning for RFID data streams. In *Proceedings of VLDB'06*, 2006.

[16] M. H. Kalos. Monte carlo methods in the physical sciences. In *Proceedings of WSC'07*, 2007.

[17] N. Khoussainova, M. Balazinska, and D. Suciu. Towards correcting input data errors probabilistically using integrity constraints. In *Proceedings of MobiDE'06*, 2006.

[18] N. Khoussainova, M. Balazinska, and D. Suciu. PEEX: Extracting Probabilistic Events from RFID Data. In *Proceedings of ICDE'08*, 2008.

[19] A. Krohn, T. Zimmer, M. Beigl, and C. Decker. Collaborative Sensing in a Retail Store Using Synchronous Distributed Jam Signalling. In *Proceedings of Pervasive'05*, 2005.

[20] J. Rao, S. Doraiswamy, H. Thakkar, and L. S. Colby. A deferred cleansing method for RFID data analytics. In *Proceedings of VLDB'06*, 2006.

[21] C. Ré, J. Letchner, M. Balazinksa, and D. Suciu. Event queries on correlated probabilistic streams. In *Proceedings of SIGMOD'08*, 2008.

[22] T. Tran, C. Sutton, R. Cocci, Y. Nie, Y. Diao, and P. Shenoy. Probabilistic Inference over RFID Streams in Mobile Environments. In *Proceedings of ICDE'09*, 2009.

[23] Y.-J. Tu and S. Piramuthu. Reducing false reads in RFID-embedded supply chains. *Journal of Theoretical and Applied Electronic Commerce Research*, 2008.

[24] Y.-J. Tu, W. Zhou, and S. Piramuthu. Identifying RFID-embedded objects in pervasive healthcare applications. *Decision Support Systems*, 2009.

[25] F. Wang and P. Liu. Temporal management of RFID data. In *Proceedings of VLDB'05*, 2005.

[26] L. Weiss Ferreira Chaves, E. Buchmann, and K. Böhm. Tagmark: Reliable estimations of RFID tags for business processes. In *Proceedings of KDD'08*, 2008.

[27] E. Welbourne, N. Khoussainova, J. Letchner, Y. Li, M. Balazinska, G. Borriello, and D. Suciu. Cascadia: A System for Specifying, Detecting, and Managing RFID Events. In *Proceedings of MobiSys'08*, 2008.

[28] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. 2005.

[29] J. Xie, J. Yang, Y. Chen, H. Wang, and P. S. Yu. A sampling-based approach to information recovery. In *Proceedings of ICDE'08*, 2008.